RSAC 2025 Conference

Many Voices. One Community.

SESSION ID: IAIS-W01

Data-as-Code: Securing the Future of AI Model Development

Bharath Chandrasekhar, Sr. Staff Software Engineer, Google
Anton Chuvakin, Sr. Staff Security Advisor, Google

Disclaimer

Presentations are intended for educational purposes only and do not replace independent professional judgment. Statements of fact and opinions expressed are those of the presenters individually and, unless expressly stated to the contrary, are not the opinion or position of RSA Conference LLC or any other co-sponsors. RSA Conference LLC does not endorse or approve, and assumes no responsibility for, the content, accuracy or completeness of the information presented.

Attendees should note that sessions may be audio- or video-recorded and may be published in various media, including print, audio and video formats without further notice. The presentation template and any media capture are subject to copyright protection.

© 2025 RSA Conference LLC or its affiliates. The RSAC and RSAC CONFERENCE logos and other trademarks are proprietary. All rights reserved.

GenAI

Enables wonderful experiences

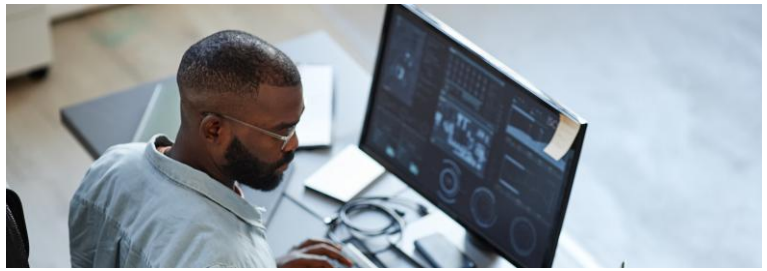
- Understands natural language
- Generates realistic looking content
- Follows instructions
- General purpose and multi-domain
- Learns new tasks using only a few examples
- Augments, assists, and automates

… and comes with risks

- Privacy concerns, Lacks explainability
- Regurgitation, Harmful content
- Prompt attacks
- Hallucinations, Bias
- Data poisoning
- Unpredictability, Large blast radius

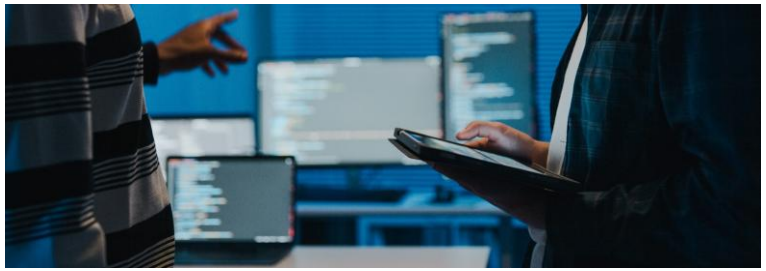
- Governance
- Explainability
- Security
- Privacy
- Ethics
- Safety
- BYO-AI
- Compliance
- IP
- Regulatory

Secure AI Framework (SAIF)



1 Leverage and extend to ML Google's secure-by-default expertise and infrastructure

2 Detect and respond in time



3 Automate defenses, using AI where relevant

4 Harmonize protections across platforms & tools

5 Adapt protections continuously

6 Contextualize AI system risks in surrounding business processes

Layers of Securing AI

Data

Protect the data used to train and fine tune the model and the data used for GenAI capabilities such as RAG

Model

Protect the model itself and the process by which it is developed

Application

Protect application leveraging GenAI, whether that GenAI capability is customer-, Google-, or vendor-managed

Infrastructure

Protect the infrastructure and platforms underpinning GenAI applications, enabling compliance and infrastructure control

RSAC 2025 Conference

Lessons from 40+ years of software development

Secure the Code

1. Version Control Systems
2. Build Systems
3. Distribution and Deployment

Secure the Config

IaC brought the same learnings to Config

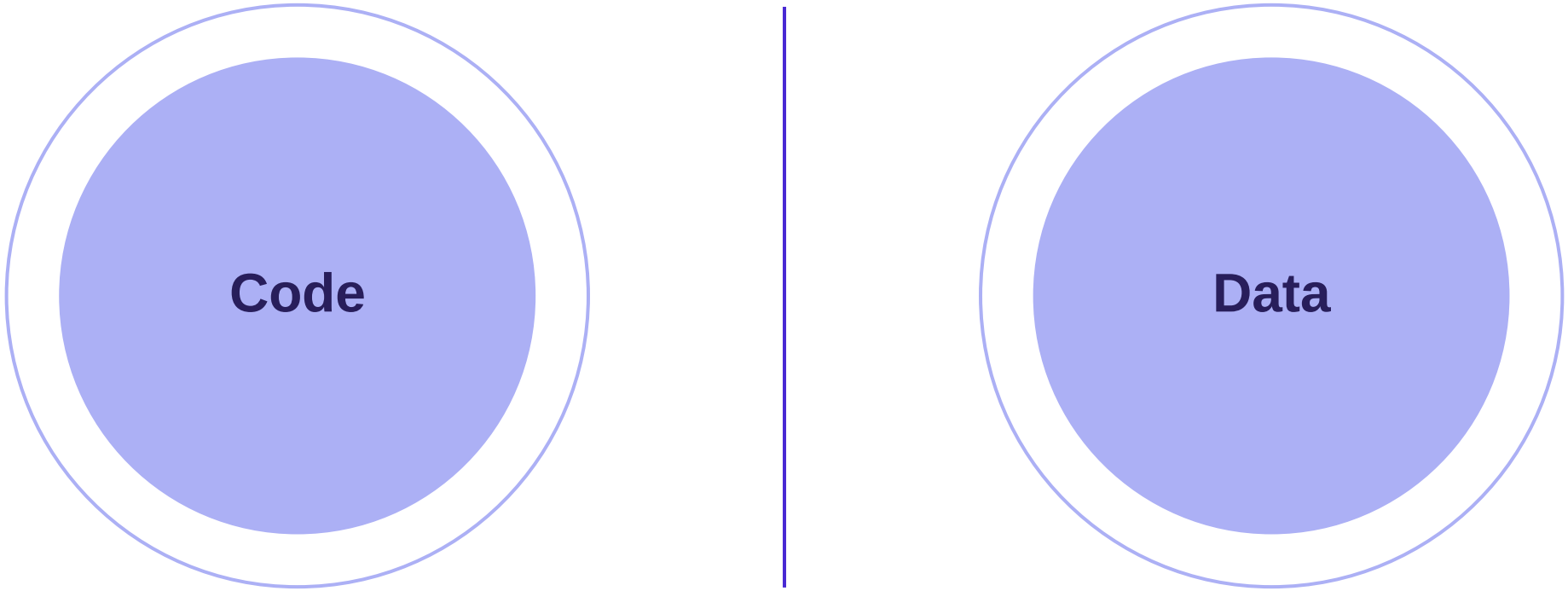
Secure, version controlled infrastructure config alongside code

- Validated provenance and lineage of each change
- Signed commits
- Tamper-proof storage model with strong access controls

Secure deployment and distribution

- (Mostly) easy rollbacks

Where are the AI models?



This is what you write

This is what is being processed

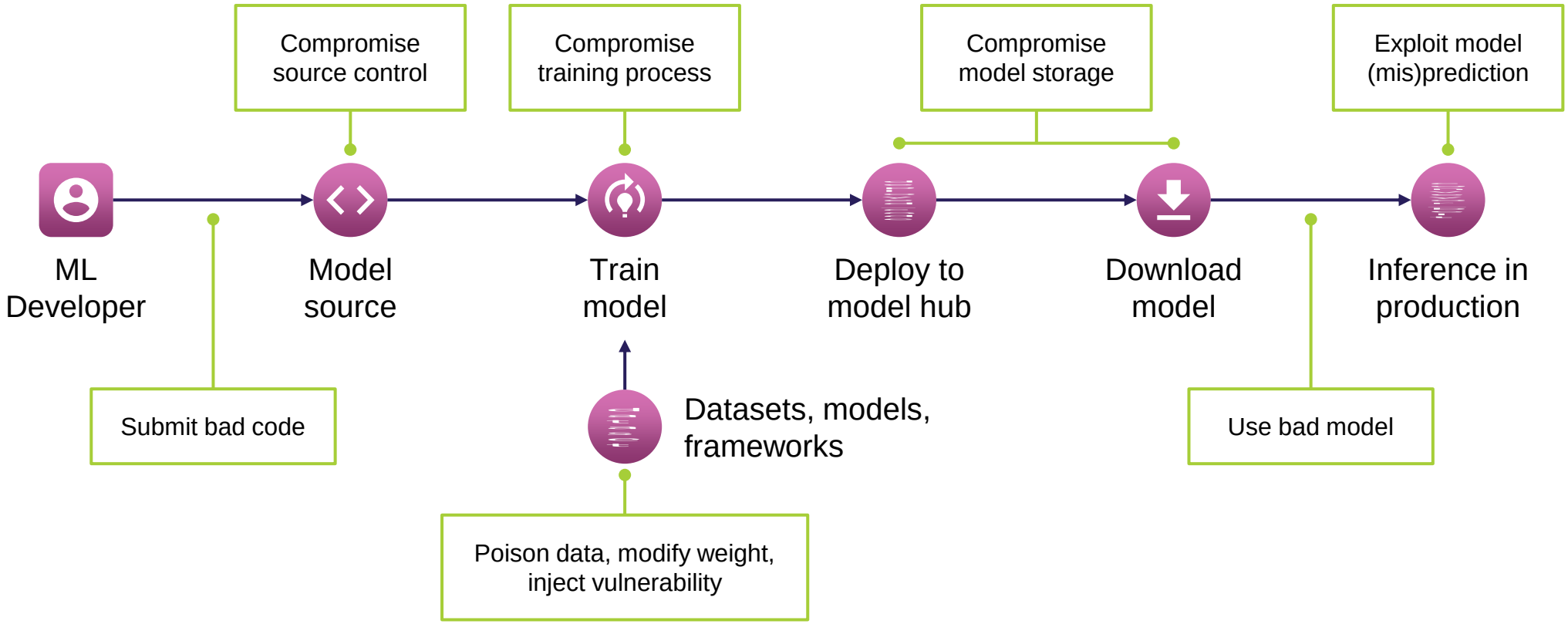
Enter AI…

Data is a critical dependency along with code and config

Data is everywhere; model training, fine tuning, and even serving

Data is an integral part of the trained model

Potential impact of data compromise on AI Supply Chain



Source: https://storage.googleapis.com/gweb-research2023-media/pubtools/7769.pdf

And not all data is created equal

1. Do you know the provenance of the data? Have you vetted it?
2. Is there PII, PHI or copyrighted material in the dataset?
3. Is there a possibility of the dataset being subtly poisoned?
4. If you're using synthetic data, is it inaccurate or biased?
5. Are you complying with data residency regulations when storing and processing data?

AI Data Challenges: Consequences

Training cycles are expensive.

Potential cost: $xxM

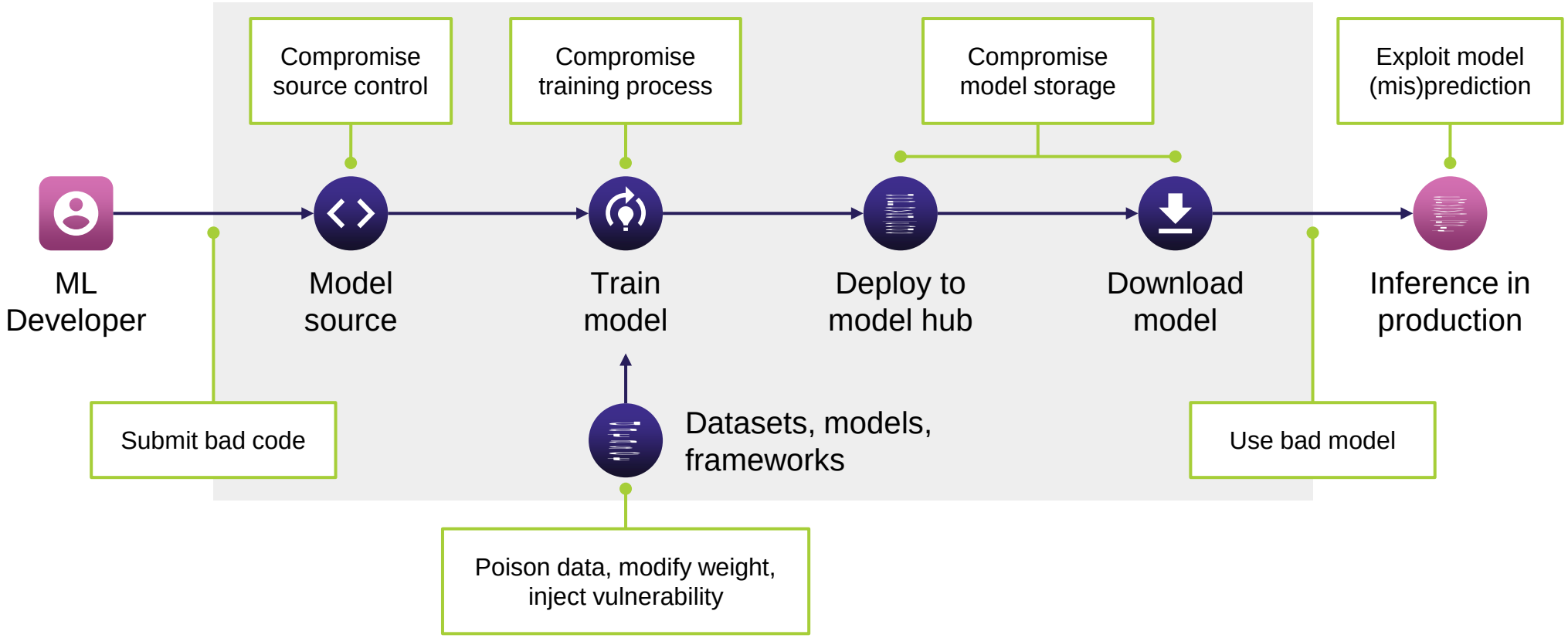
Scale of data:

Human review is almost impossible. Imagine sifting through petabytes of data and unbounded synthetic data.

Regulatory failures:

Might force rollback and a long rebuild cycle

Learn from the past - treat Data as Code



Data as Code Tenets

1

Provenance and lineage

2

Secure the data usage

3

Treat model weights as data:

Secure at rest

4

Apply governance:

Entire lifecycle

Approach used at Google***

Used tried and tested tools; adapt them for data

Some of them include:

- BCID / Binary Authorization / SigStore
- Index / Search / CNS: For granular cleaning, versioning, storage
- KMS / Secret Manager
- Runtime protection of model training frameworks

Provenance and lineage

1

Version and track provenance/lineage of data and models

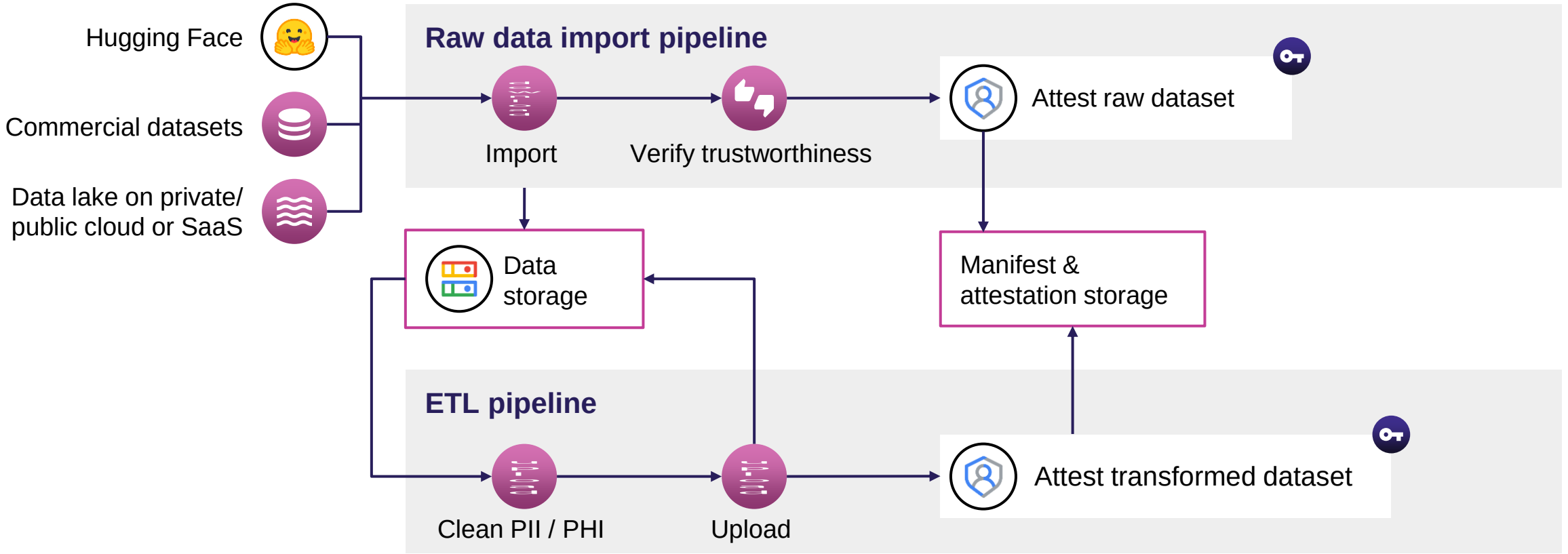
Store each checkpoint and version of a dataset/ model as an immutable, verifiable artifact

Address dataset size with newer data-focused tools

- DVC (git inspired)
- OCI Artifact Registries (Docker inspired)

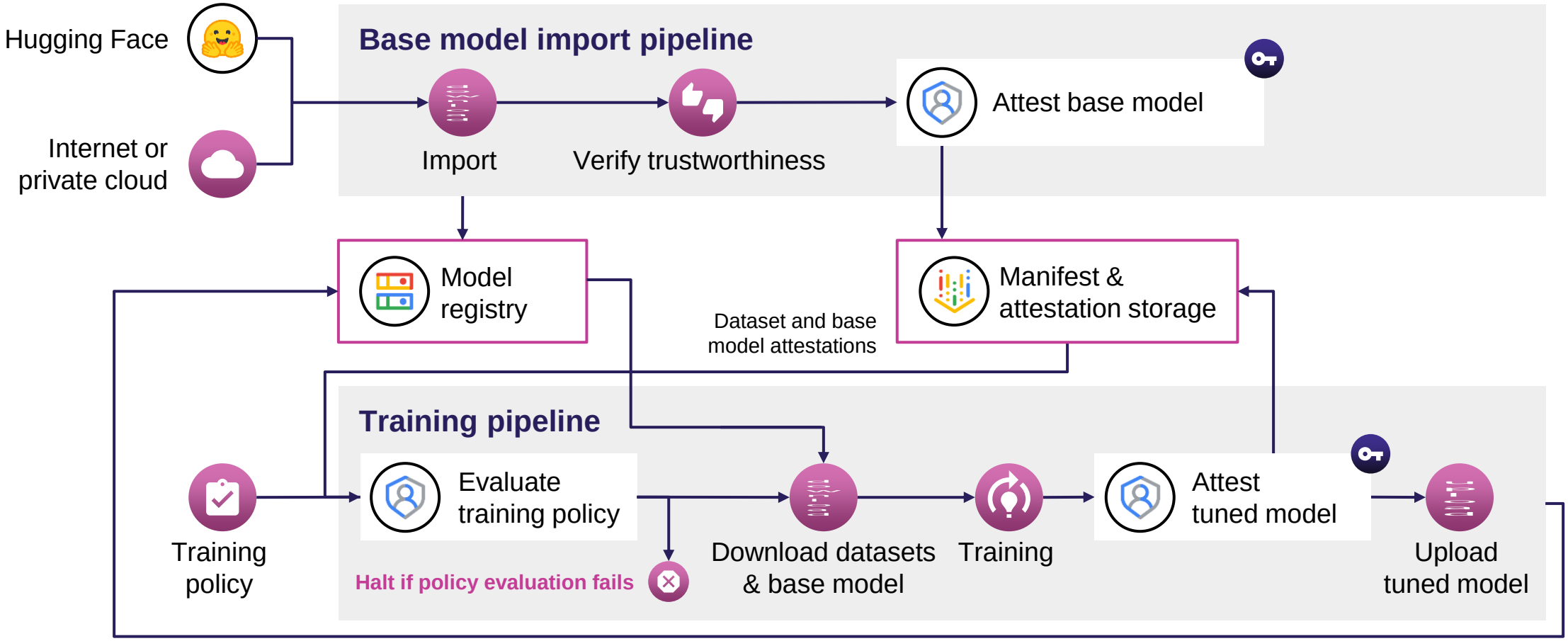
Secure the data usage

2



Treat model weights as data: secure at training

3



Treat model weights as data: secure at rest

3

Protect data, models and their metadata against tampering at rest

- Use trusted build process that generates attestations (signatures) covering dataset/model contents, lineage and metadata (including any custom metadata provided by the caller)
- Apply strong access controls on dataset/ model repository
- Verify attestations before use for training, fine-tuning or inference

Govern the use of specific kinds of data for training, to preserve desired level of privacy, factuality and regulatory compliance?

- Verify dataset attestations and metadata before training

Apply governance: Pre-deployment

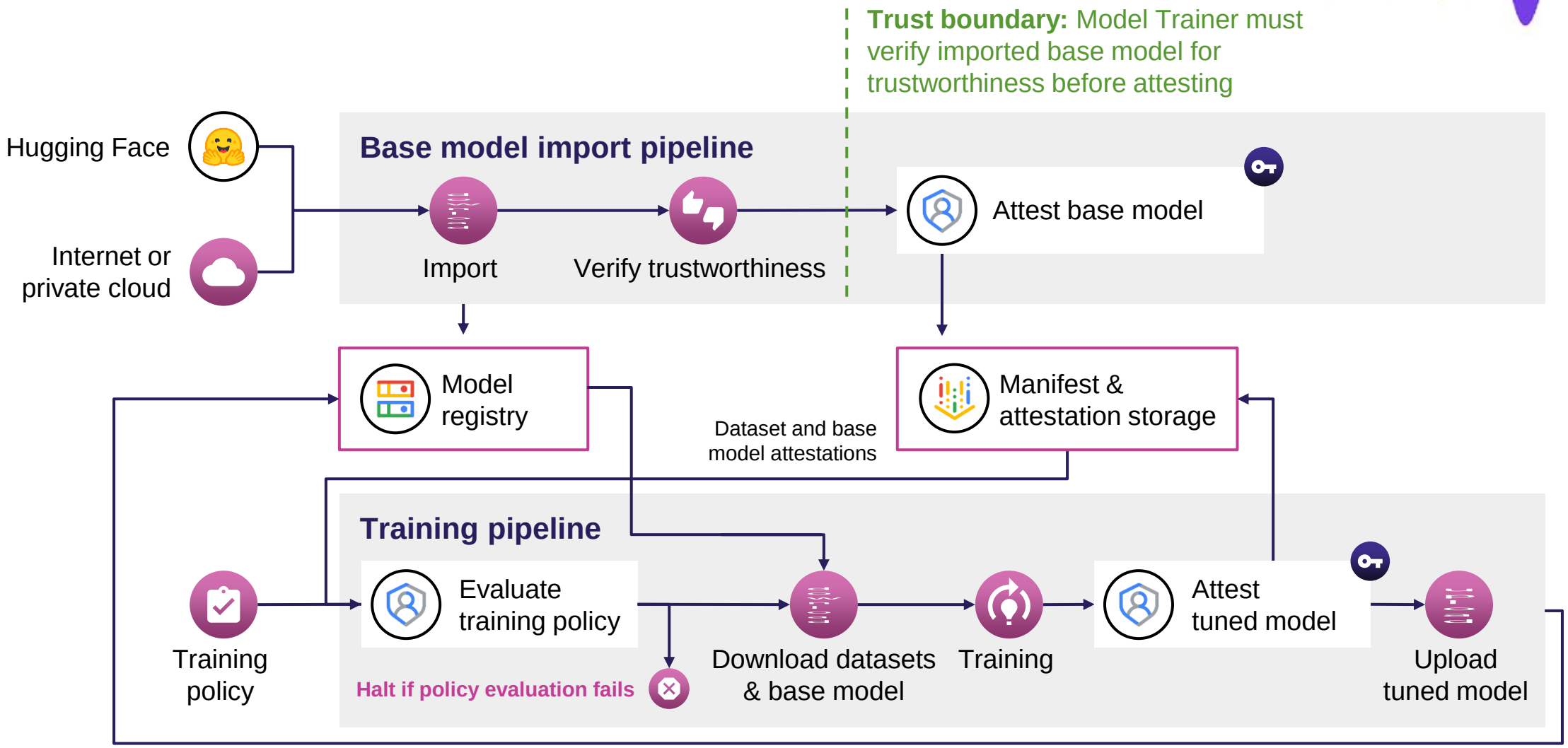
4

Trigger a policy check

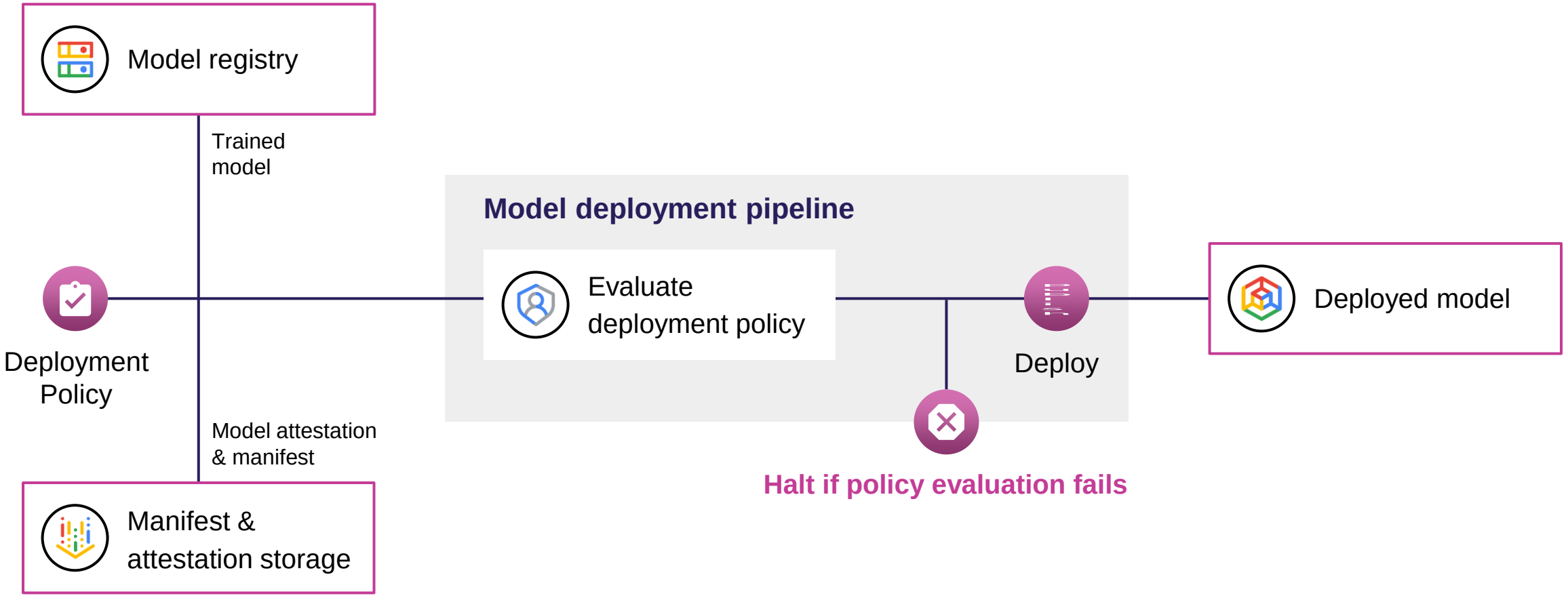
Verify dataset/model attestation, properties at the beginning of every pipeline that performs **a data transformation, training or model deployment**

Incorporate admissions control for any deployment

Example: Model fine-tuning



Example: Model deployment



DaC vs IaC

Feature	Infrastructure as Code (IaC)	Data as Code
Core Focus	Managing and provisioning infrastructure (servers, networks, etc.) [1]	Managing and securing data used in AI/ML systems [2]
Primary Assets	Configuration files, templates (e.g., Terraform, CloudFormation) [1]	Datasets, model weights, data pipelines, and related metadata [3]
Key Principles	Version control, automation, repeatability, idempotency [1]	Provenance and lineage, secure usage, treating model weights as data, governance [2]
Similarities	Both apply software engineering practices to traditionally less "code-like" domains.	Both apply software engineering practices to traditionally less "code-like" domains.

Conclusion

1. Data is a first-class input into AI systems, and it must be treated as code.
2. Data as Code (DaC) provides lineage, integrity, and safety of AI models.
3. DaC means applying best practices from software development to data.
4. Google has used DaC to successfully secure AI development.
5. Adopt DaC to start securing the future of your AI development.

Apply What You Have Learned Today

Next week you should:

- Create or refine data inventory & generation processes (start to!)
- Catalog AI use cases and relevant data

In the first three months following this presentation you should:

- Create or refine processes for securing ML pipelines and data
- Integrate data governance processes with AI needs

Within six months you should:

- Arrive at an accurate and “live” view of data used for AI

Discussion

1. What do you think of “data as code” concept?
2. Where does AI intersect with data security, in your view?
3. What do we need to do differently in data security due to AI?
4. What is your top AI data security concern?

RSAC™ 2025 Conference

Many Voices.
One Community.

Thank you